

## Amendment to the Claims

### Listing of Claims:

1. (Currently amended) A method for restoring persistently stored objects of an object-oriented environment established in a computer system having a volatile memory and a persistent storage, the method comprising the steps of:

retrieving from said persistent storage a first list comprising first references to segments, stored in said persistent storage;

retrieving all segments referenced by said first references and storing them in said volatile memory;

saving in said first list the difference between an old memory address, at which the segment used to reside in the volatile memory, and a new memory address at which said segment is stored;

retrieving from said persistent storage a second list comprising second references to blocks, whereby one or more of said blocks contain an object description;

determining which segment contains said block referenced by a particular element of said second list;

creating ~~an~~ a new object in said volatile memory using said object description from said segment and saving a new address of said ~~created~~ new object in said second list in volatile memory;

initializing said new object with values taken from said object description; and

determining said new addresses of ~~said new object~~ objects referenced by the ~~newly-created~~ object and setting said new address as the reference in said new object.

2. (Previously presented) The method according to claim 1, whereby said first list and/or said second list are ordered lists.

3. (Previously presented) The method according to claim 1 or 2, whereby said first list and/or said second list are organized as a B-tree.

4. (Previously presented) The method according to claim 1, whereby the elements of said first ordered list are indexed by said first references.

5. (Previously presented) The method according to Claim 1 one of the preceding claims, whereby each of said first references corresponds to the old memory address at which the respective segment used to reside in the volatile memory.

6. (Previously presented) The method according to Claim 1, whereby the elements of said second ordered list are indexed by said second references.

7. (Previously presented) The method according to one Claim 1, whereby each of said second references corresponds to the old memory address at which the respective block used to reside in said volatile memory.

8. (Previously presented) The method according to Claim 1, whereby said object

description is formed by a collection of values owned by an object for the variables belonging to its class.

9. (Previously presented) The method according to Claim 1, whereby for each value in said object description of variables having a variable length the method further comprising the steps of:

allocating a number of blocks that allows to keep the actual value of the variable having variable length;

creating a linked list of said number of blocks;

saving said value into said number of blocks; and

storing a reference to the head of the linked list in said object description.

10. (Previously presented) The method according to claim 1, whereby determining the segment that contains said block referenced by a particular element of said second list comprises the step of searching in said first ordered list (segment map) for the segment that contains said portion of said segment (block) referenced by said element.

11. (Previously presented) The method according to claim 1, whereby determining the segment that contains said block referenced by a particular element of said second list further comprises the step of calculating the new address by adding the reference to said block (that corresponds to the old memory address) and said difference between said old memory address and said new memory address.

12. (Previously presented) The method according to claim 1, whereby determining the new addresses of objects referenced by the newly created object comprises the step of searching in said second list (object map) for the element said contains the new address of the referenced object, that is referenced by the old address of the respective object description.

13. (Previously presented) The method according to claim 1, whereby for all references to heads of linked lists the method further comprising the steps of:

reading all blocks of said linked list;

allocating memory to store the value of the variable retrieved from the linked list;

and

storing the value in said allocated memory.

14. (Currently amended) A method for persistently storing objects of an object-oriented environment established on a computer system having a volatile memory and a persistent storage, the method comprising steps of:

allocating segments in said volatile memory; ~~segments~~;

creating a first list comprising first references to said segments;

creating a second list comprising second references to blocks, wherein the blocks are portions of said segments;

allocating a block of one of said segments,

creating ~~an~~ a first object description for ~~an~~ a first object by saving values owned by the first object of the variables belonging to its class into said allocated block and saving a new address of said first object in the second list;

adding a new element to said second list containing ~~the~~ a particular reference to said first object description;

determining the address of ~~another~~ a second object description of ~~another~~ a second object referenced in said first object;

setting the address of said respective object description as the reference in the created object description;

storing said second list on said persistent storage;

storing the segments referenced by said first list on said persistent storage; and

storing said first list on said persistent storage.

15. (Previously presented) The method according to claim 14, whereby said first list and/or said second list are ordered lists.

16. (Previously presented) The method according to claim 14, whereby said first list and/or said second list are organized as a B-tree.

17. (Previously presented) The method according to claim 14, whereby the elements of said first ordered list are indexed by said first references.

18. (Previously presented) The method according to claim 14, whereby each of said first references corresponds to the current memory address at which the respective segment resides in the volatile memory.

19. (Previously presented) The method according to claim 14, whereby the elements of said second ordered list are indexed by said second references.

20. (Previously presented) The method according to claim 14, whereby each of said second references corresponds to the current memory address at which the respective block resides in said volatile memory.

21. (Previously presented) The method according to claim 14, whereby determining the address of the object description of another object referenced in said object comprises the step of searching in said second ordered list for the element said contains the address of the object description of the referenced other object.

22. (Previously presented) The method according to claim 14, whereby determining the address of the object description of another object referenced in said object comprises a step of using a reference to the respective object description provided by each object.

23. (Previously presented) The method according to claim 14, whereby for each value of variables comprises a variable length the method further comprises steps of:

allocating a number of portions of one of said pieces of memory that allows to keep the actual value of the variable length variable;

creating a linked list of said number of portions;

saving value into said number of portions; and

storing a reference to the head of the linked list in said object description.

24. (Previously presented) A computer program product stored on a computer usable medium, comprising computer readable program instructions for:

- allocating in said volatile memory segments;
- creating a first list comprising first references to said segments;
- creating a second list comprising second references to blocks;
- allocating a block of one of said segments,
- creating an object description by saving values owned by the object of the variables belonging to its class into said allocated block;
- adding a new element to said second list containing the particular reference to said created object description;
- determining the address of the object description of another object referenced in said object;
- setting the address of said respective object description as the reference in the created object description;
- storing said second list on said persistent storage;
- storing the segments referenced by said first list on said persistent storage; and
- storing said first list on said persistent storage.